



STUDENT ID NO

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 3, 2015/2016

ECP2216 – MICROCONTROLLER AND MICROPROCESSOR SYSTEMS

(All sections / Groups)

3 JUNE 2016
9.00 a.m. – 11.00 a.m.
(2 Hours)

INSTRUCTIONS TO STUDENTS

1. This Question paper consists of 10 pages with 5 Questions only.
 2. Attempt **ALL FIVE COMPULSORY** questions. All questions carry equal marks and the distribution of the marks for each question is given.
 3. Please write all your answers in the Answer Booklet provided.

Question 1

- (a) Convert BD_{16} into 8-bit two's complement number and identify it is a positive or negative number.
[2 marks]
- (b) A memory block has 20 address lines and 32 data lines. Express the memory capacity of this memory block in the units of Bytes.
[3 marks]
- (c) Von Neumann architecture is a computer architecture described by the mathematician and physicist John von Neumann in 1945. Illustrate by sketching a block diagram of this architecture.
[5 marks]
- (d) (i) Define the term ***Microarchitecture***.
[3 marks]
- (iii) The 80386DX includes six functional units that operate in parallel for pipelined processing. Name these six functional units and highlight which unit provides memory management in protected mode.
[7 marks]

Continued ...

Question 2

- (a) Identify the bit addresses which are set to 1 after the execution of the following instruction.

MOV 20H, # 54H

[3 marks]

- (b) Determine the contents of Program Status Word (PSW) and Accumulator (A) after the execution of the following instruction sequence.

(Assume initial value: A=00H and PSW=00H)

**MOV A, #0C8H
MOV R7, #58H
ADD A, R7**

[4 marks]

- (c) Assume that the available memory ICs are 1Kbytes ROM and 1Kbytes RAM. Design an 8051 microcontroller based system that can address contiguous 4Kbytes of memory space. 2Kbytes of RAM should occupy the first portion of the memory space followed by 2Kbytes of ROM. (*Draw and label the system configuration showing the 8051 signal lines to be used for data, address and control buses.*)

[13 marks]

Continued ...

Question 3

- (a) State any **THREE** available addressing modes for MCS-51 program branching instructions.
- [3 marks]
- (b) Determine the contents of the accumulator (ACC) and PSW register after the execution of **EACH** instruction in the following MCS-51 assembly language program. Assume initial value of PSW is 00H.

```
ORG 0000H
MOV A, #26H
ADD A, #0FFH
SUBB A, #16H
END
```

- (c) An MCS-51 assembly language subroutine is shown as following:
- [6 marks]

| | |
|-------------|----------------|
| SUBROUTINE: | MOV R6, #200 |
| AGAIN: | NOP |
| | NOP |
| | DJNZ R6, AGAIN |
| | RET |

- (i) Assume that a 12 MHz crystal frequency is used, calculate the total execution time of the subroutine.
- [3 marks]
- (ii) Using the MCS-51 Opcode Map, convert the instruction

“ DJNZ R6, AGAIN ”

into the corresponding machine code. Assume the first instruction of the subroutine is addressed at **0000H**.

[5 marks]

- (iii) Modify the subroutine to increase the total execution time to 0.8 seconds.
- [3 marks]

Continued ...

Question 4

- (a) State **ALL** available interrupt sources in an 8051 microcontroller and arrange them in the order corresponding to their default priority.
[5 marks]
- (b) Assume 11.0592MHz crystal frequency, 8-bit data, 1 stop bit, no parity and operation at 9600 baud rate generated by Timer 1. Write a MCS-51 assembly language subroutine to receive a character from serial port and store the character in R4. (*Show the initialization of SCON, TMOD and TH1 registers.*)
[5 marks]
- (c) An 8051 microcontroller system with INT1 pin and INT0 pin connected to a switch that is normally high. Write a MCS-51 assembly language program to perform the following tasks.

INT0 pin goes low: Use timer 0 interrupt to generate a pulse width of 1ms from P1.0.

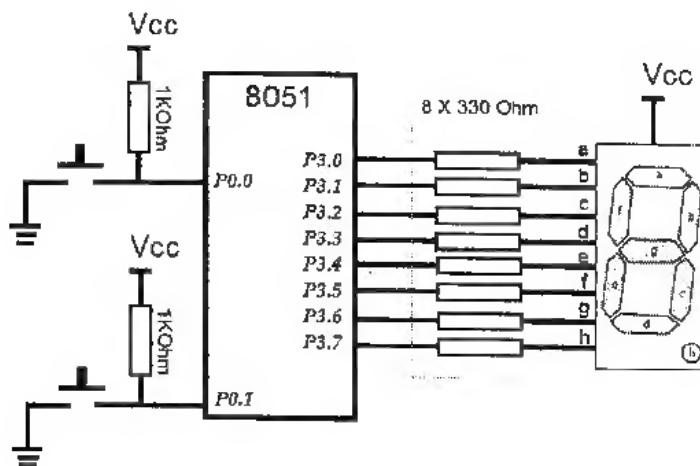
INT1 pin goes low: Use timer 1 interrupt to generate a pulse width of 0.5ms from P1.1.

[10 marks]

Continued ...

Question 5

- (a) Figure 5(a) depicts an 8051 microcontroller interfaces to a common anode-type seven-segment LED display device on Port 3 and two press buttons on P0.0 and P0.1.

**Figure 5(a)**

- (i) Fill in Table 5(a) to define the bit patterns for each character to decode the seven-segment LED display.

Table 5(a)

| | P3.7 | P3.6 | P3.5 | P3.4 | P3.3 | P3.2 | P3.1 | P3.0 |
|------------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| Character | h | g | f | e | d | c | b | a |
| E | | | | | | | | |
| C | | | | | | | | |
| P | | | | | | | | |
| A | | | | | | | | |

[2 marks]

- (ii) Assume 1 second delay subroutine `DELAY` is available. Write a MCS-51 assembly language program that will wait for the button press on P0.0. Once the button is pressed, seven-segment LED display will repeatedly display the characters in sequence starting from E, C, P and A. Time duration for each character to be displayed is 1 second. The display will be stopped only if the button press on P0.1 occurs.

[8 marks]

Continued ...

- (b) An 8051 microcontroller based automated coffee maker machine is shown in Figure 5(b) which performs the following process:

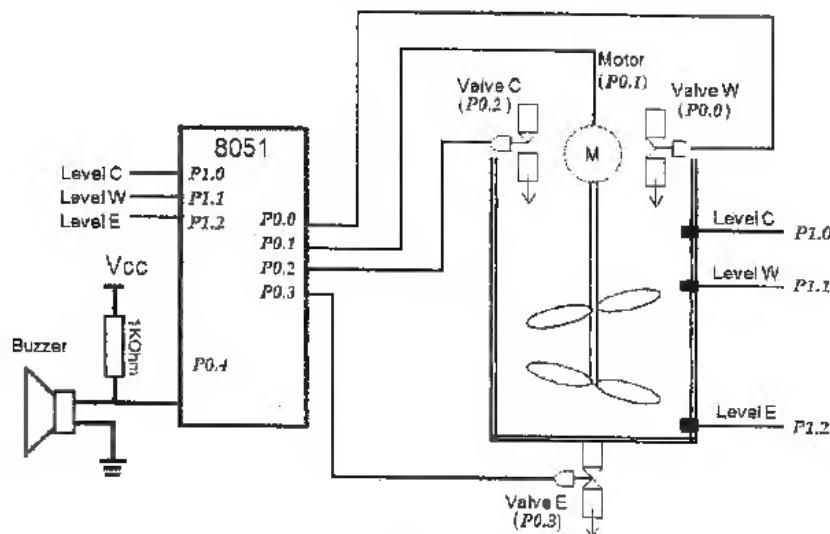


Figure 5(b)

1. The chamber is first filled with hot drinking water through a solenoid Valve W.
2. When the hot drinking water reaches Level W, Valve W is closed and the chamber is now filled with coffee powder through Valve C.
3. When the mixture in the chamber reaches Level C, Valve C is closed.
4. The mixer motor starts the stirring process that last for approximately 2 minutes.
5. After that, the drainage Valve E opens to dispense the mixture.
6. When the mixture reaches Level E, Valve E is closed and the buzzer will sound for approximately 1 minute to indicate the completion.
7. The whole process is repeated from Step 1 again.

The chamber has three level sensors that send signals to input lines **P1.0** to **P1.2**. A logical **HIGH** from the sensor indicates that the level has been reached. The output lines **P0.0**, **P0.2**, and **P0.3** provide signals to the solenoid valves. A logical **HIGH** from the lines will open the corresponding valve. The output lines **P0.1** and **P0.4** provide signals to the mixer motor and buzzer respectively which are both activated by a logical **HIGH**. Write a MCS-51 assembly language program to carry out the process. Assume 12MHz crystal frequency is used

[10 marks]

End of Page

APPENDIX

Special Function Register Formats

Interrupt Enable (IE)

| Bit Addr. | AFH | - | - | ACH | ABH | AAH | A9H | A8H |
|-----------|-----|---|---|-----|-----|-----|-----|-----|
| Name | EA | - | - | ES | ET1 | EX1 | ET0 | EX0 |

| BIT | SYMBOL | FUNCTION (Enable=1, Disable=0) |
|------|--------|---|
| IE.7 | EA | Global enable/disable. EA = 1, each individual source is enabled/disabled by setting/clearing its enable bit. EA = 0, disable all interrupts. |
| IE.6 | - | Undefined |
| IE.5 | - | Not implemented in 8051. ET2 for 8052. |
| IE.4 | ES | Serial port interrupt enable bit. |
| IE.3 | ET1 | Timer 1 interrupt enable bit. |
| IE.2 | EX1 | External interrupt enable bit. |
| IE.1 | ET0 | Timer 0 interrupt enable bit. |
| IE.0 | EX0 | External interrupt enable bit |

Interrupt Priority (IP)

| Bit Addr. | - | - | - | BCH | BBH | BAH | B9H | B8H |
|-----------|---|---|---|-----|-----|-----|-----|-----|
| Name | - | - | - | PS | PT1 | PX1 | PT0 | PX0 |

| BIT | SYMBOL | FUNCTION (Enable=1, Disable=0) |
|------|--------|--|
| IP.7 | - | Undefined |
| IP.6 | - | Undefined |
| IP.5 | - | Not implemented in 8051. PT2 for 8052. |
| IP.4 | PS | Serial port interrupt priority bit. |
| IP.3 | PT1 | Timer 1 interrupt priority bit. |
| IP.2 | PX1 | External interrupt priority bit. |
| IP.1 | PT0 | Timer 0 interrupt priority bit. |
| IP.0 | PX0 | External interrupt priority bit. |

Interrupt Vectors

| Interrupt Source | Flag | Vector Address |
|------------------|-------------|----------------|
| System Reset | RST | 0000H |
| External 0 | IE0 | 0003H |
| Timer 0 | TF0 | 000BH |
| External 1 | IE1 | 0013H |
| Timer 1 | TF1 | 001BH |
| Serial Port | RI & TI | 0023H |
| Timer 2 (8052) | TF2 or EXF2 | 002BH |

Program Status Word (PSW)

| | | | | | | | | |
|-----------|-----|-----|-----|-----|-----|-----|---|-----|
| Bit Addr. | D7H | D6H | D5H | D4H | D3H | D2H | - | D0H |
| Name | CY | AC | F0 | RS1 | RS0 | OV | - | P |

Serial Control (SCON)

| | | | | | | | | |
|-----------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit Addr. | 9FH | 9EH | 9DH | 9CH | 9BH | 9AH | 99H | 98H |
| Name | SM0 | SM1 | SM2 | REN | TB8 | RB8 | TI | RI |

| BIT | SYMBOL | FUNCTION |
|--------|--------|--|
| SCON.7 | SM0 | Serial port mode bit 0 (see Table A.1). |
| SCON.6 | SM1 | Serial port mode bit 1 (see Table A.1). |
| SCON.5 | SM2 | Serial port mode bit 2; enables multiprocessor communications in modes 2 and 3; RI will not be activated if received 9 th bit is 0. In mode 1, if SM2 = 1, then RI will be activated only if a valid stop bit was received. In mode 0, SM2 should be 0. |
| SCON.4 | REN | Receiver enable; must be set to receive characters. |
| SCON.3 | TBB | Transmit bit 8; 9 th bit transmitted in modes 2 and 3; set/cleared by software. |
| SCON.2 | RB8 | Receive bit 8; 9 th bit received. |
| SCON.1 | TI | Transmit interrupt flag; set at end of character transmission; cleared by software. |
| SCON.0 | RI | Receive interrupt flag; set at end of character reception; cleared by software. |

Table A.1 The 8051 Serial Port Mode Selection

| SM0 | SM1 | Mode | Description | Baud Rate |
|-----|-----|------|----------------|-----------|
| 0 | 0 | 0 | Shift register | Fixed |
| 0 | 1 | 1 | 8-bit UART | Variable |
| 1 | 0 | 2 | 9-bit UART | Fixed |
| 1 | 1 | 3 | 9-bit UART | Variable |

Timer Control (TCON)

| | | | | | | | | |
|-----------|-----|-----|-----|-----|-----|-----|-----|-----|
| Bit Addr. | 8FH | 8EH | 8DH | 8CH | 8BH | 8AH | 89H | 88H |
| Name | TF1 | TR1 | TF0 | TR0 | IE1 | IT1 | IE0 | IT0 |

| BIT | SYMBOL | FUNCTION |
|--------|--------|---|
| TCON.7 | TF1 | Timer-1 overflow flag. Set by hardware on overflow. Cleared by hardware when processor vectors to interrupt routine. Must be cleared by software when not involve interrupt |
| TCON.6 | TR1 | Timer-1 run control bit. Set/cleared by software to turn timer/counter on/off. |
| TCON.5 | TF0 | Timer-0 overflow flag. Do the same function as TF1 but for Timer-0 |
| TCON.4 | TR0 | Timer-0 run control bit. Do the same function as TR1 but for Timer-0. |
| TCON.3 | IE1 | External interrupt-1 edge flag. Set by hardware when interrupt-1 falling edge is detected. Cleared by hardware when interrupt is processed. |
| TCON.2 | IT1 | Interrupt-1 Type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts. |
| TCON.1 | IE0 | External interrupt-0 edge flag. Do the same function as IE1 but for external interrupt-0. |
| TCON.0 | IT0 | Interrupt-0 Type control bit. Do the same function as IT1 but for external interrupt-0. |

Timer Mode (TMOD)

| | | | | | | | | |
|------|------|-----|----|----|------|-----|----|----|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | GATE | C/T | M1 | M0 | GATE | C/T | M1 | M0 |

| BIT | SYMBOL | FUNCTION |
|--------|--------|--|
| TMOD.7 | GATE1 | When this bit is set the timer will only run when INT1 (P3.3) is high (hardware control). When this bit is cleared the timer will run regardless of the state of INT1 (software control). |
| TMOD.6 | C/T1 | Timer / Counter select bit. $C/T = 0 \rightarrow$ Timer operation. $C/T = 1 \rightarrow$ Counter operation. |
| TMOD.5 | M1 | Mode selection bits (see Table A.2). [for timer 1] |
| TMOD.4 | M0 | Mode selection bits (see Table A.2). [for timer 1] |
| TMOD.3 | GATE0 | Exactly the same function as GATE1 but for Timer0 |
| TMOD.2 | C/T0 | Exactly the same function as C/T1 but for Timer0 |
| TMOD.1 | M1 | Mode selection bits (see Table A.2). [for timer 0] |
| TMOD.0 | M0 | Mode selection bits (see Table A.2). [for timer 0] |

Table A.2 Timer Mode Selection

| M1 | M0 | Timer Mode | Description of Mode |
|----|----|------------|---------------------|
| 0 | 0 | 0 | 13-bit Timer |
| 0 | 1 | 1 | 16-bit Timer |
| 1 | 0 | 2 | 8-bit auto-reload |
| 1 | 1 | 3 | Split timer mode |

MCS-51 Opcode Map

| Instruction opands cycle | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|--------------------------------|-----------------------------|-------------------------|--------------------------|-------------------------|--------------------------|-------------------------|--------------------------|--------------------------|--------------------------|--------------------------|-----------------------------------|-----------------------------------|-----------------------------|-----------------------------|---------------------------|---|
| 0 NOP | 13 3B JNC bit, rel 2C | 3B JB bit, rel 2C | 2B JNC bit, rel 2C | 2B JC bit, rel 2C | 2B JNZ bit, rel 2C | 2B JZ bit, rel 2C | 2B SJMP rel 2C | 2B SJMP rel 2C | 2B MON bit, rel 2C | 2B ORL bit, rel 2C | 2B ANL bit, rel 2C | 2B PUSH bit 2C | 2B POP bit 2C | 2B MOVA bit, rel 2C | 2B MOVB bit, rel 2C | |
| 1 AJMP (P0) | 25 2B ACALL (P0) | 2B AJMP (P1) | 2B ACALL (P1) | 2B AJMP (P2) | 2B ACALL (P2) | 2B AJMP (P3) | 2B ACALL (P3) | 2B AJMP (P4) | 2B ACALL (P4) | 2B AJMP (P5) | 2B ACALL (P5) | 2B AJMP (P6) | 2B ACALL (P6) | 2B AJMP (P7) | 2B ACALL (P7) | |
| 2 LJMP addr16 | 33 2B LCALL addr16 | 2B RET | 2B RETI | 2B ORL dir, A 1C | 2B ORL dir, A 1C | 2B ORL dir, A 1C | 2B XRL dir, A 1C | 2B ANL dir, A 1C | 2B ANL dir, A 1C | 2B ANL dir, A 1C | 2B MOV bit, C 2C | 2B MOV bit, C 2C | 2B CPL bit 1C | 2B CLR bit 1C | 2B MOVA bit, rel 2C | |
| 3 RR | 13 1B RRC | 1B RL | 1B RLC | 1B QRL | 1B QRL | 1B JMP | 1B MOVC bit+DTR 2C | 1B MOVC bit+DTR 2C | 1B INC bit+DTR 2C | 1B INC bit+DTR 2C | 1B MUL AB 4C | 1B SWAP A 4C | 1B SETB C 1C | 1B SETB C 1C | 1B MOVX @R0, A 2C | |
| 4 INC | 13 1B DEC | 2B ADD | 2B ADDC | 2B ORL dir, A 1C | 2B XRL dir, A 1C | 2B XRL dir, A 1C | 2B DIV dir, A 1C | 2B DIV dir, A 1C | 2B DIV dir, A 1C | 2B DIV dir, A 1C | 2B CJNE A, @data, rel 4C | 2B SWAP A 4C | 1B CLR A 1C | 1B CLR A 1C | 1B MOVX @R1, A 2C | |
| 5 INC | 23 2B DEC | 2B ADD | 2B ADDC | 2B ORL dir, A 1C | 2B XRL dir, A 1C | 2B XRL dir, A 1C | 2B MOV dir, A 1C | 2B MOV dir, A 1C | 2B MOV dir, A 1C | 2B SUBB dir, A 1C | 2B CJNE A, dir, rel 2C | 2B CJNE A, dir, rel 2C | 2B DJNZ dir, rel 2C | 2B MOV dir, A 1C | 2B CPL A 1C | |
| 6 INC | 13 1B DEC | 1B ADD | 1B ADDC | 1B ORL dir, A 1C | 1B XRL dir, A 1C | 1B XRL dir, A 1C | 1B MOV dir, A 1C | 1B MOV dir, A 1C | 1B MOV dir, A 1C | 1B SUBB dir, A 1C | 1B CJNE A, @data, rel 2C | 1B CJNE A, @data, rel 2C | 1B XCHD A, @R0, tc 1C | 1B XCHD A, @R0, tc 1C | 1B MOV @R0, A 1C | |
| 7 INC | 13 1B DEC | 1B ADD | 1B ADDC | 1B ORL dir, A 1C | 1B XRL dir, A 1C | 1B XRL dir, A 1C | 1B MOV dir, A 1C | 1B MOV dir, A 1C | 1B MOV dir, A 1C | 1B SUBB dir, A 1C | 1B CJNE A, @data, rel 2C | 1B CJNE A, @data, rel 2C | 1B XCHD A, @R1, tc 1C | 1B XCHD A, @R1, tc 1C | 1B MOV @R1, A 1C | |
| 8 INC | 18 1B DEC | 1B ADD | 1B ADDC | 1B ORL dir, A 1C | 1B XRL dir, A 1C | 1B XRL dir, A 1C | 1B MOV dir, A 1C | 1B MOV dir, A 1C | 1B MOV dir, A 1C | 1B SUBB dir, A 1C | 1B CJNE A, R0, tc 1C | 1B CJNE A, R0, tc 1C | 1B DJNZ R0, rel 2C | 1B MOV A, R0 1C | 1B MOV R0, A 1C | |
| 9 INC | 18 1C R1 | 1B DEC | 1B ADD | 1B ADDC | 1B ORL dir, A 1C | 1B XRL dir, A 1C | 1B XRL dir, A 1C | 1B MOV dir, A 1C | 1B MOV dir, A 1C | 1B SUBB dir, A 1C | 1B CJNE A, R1, dir 2C | 1B CJNE A, R1, dir 2C | 1B XCHD A, R1, tc 1C | 1B XCHD A, R1, tc 1C | 1B MOV R1, A 1C | |
| A INC | 18 1C R2 | 1B DEC | 1B ADD | 1B ADDC | 1B ORL dir, A 1C | 1B XRL dir, A 1C | 1B XRL dir, A 1C | 1B MOV dir, A 1C | 1B MOV dir, A 1C | 1B SUBB dir, A 1C | 1B CJNE A, R2, dir 2C | 1B CJNE A, R2, dir 2C | 1B XCHD A, R2, tc 1C | 1B XCHD A, R2, tc 1C | 1B MOV R2, A 1C | |
| B INC | 18 1C R3 | 1B DEC | 1B ADD | 1B ADDC | 1B ORL dir, A 1C | 1B XRL dir, A 1C | 1B XRL dir, A 1C | 1B MOV dir, A 1C | 1B MOV dir, A 1C | 1B SUBB dir, A 1C | 1B CJNE A, R3, dir 2C | 1B CJNE A, R3, dir 2C | 1B XCHD A, R3, tc 1C | 1B XCHD A, R3, tc 1C | 1B MOV R3, A 1C | |
| C INC | 18 1C R4 | 1B DEC | 1B ADD | 1B ADDC | 1B ORL dir, A 1C | 1B XRL dir, A 1C | 1B XRL dir, A 1C | 1B MOV dir, A 1C | 1B MOV dir, A 1C | 1B SUBB dir, A 1C | 1B CJNE A, R4, dir 2C | 1B CJNE A, R4, dir 2C | 1B XCHD A, R4, tc 1C | 1B XCHD A, R4, tc 1C | 1B MOV R4, A 1C | |
| D INC | 18 1C R5 | 1B DEC | 1B ADD | 1B ADDC | 1B ORL dir, A 1C | 1B XRL dir, A 1C | 1B XRL dir, A 1C | 1B MOV dir, A 1C | 1B MOV dir, A 1C | 1B SUBB dir, A 1C | 1B CJNE A, R5, dir 2C | 1B CJNE A, R5, dir 2C | 1B XCHD A, R5, tc 1C | 1B XCHD A, R5, tc 1C | 1B MOV R5, A 1C | |
| E INC | 18 1C R6 | 1B DEC | 1B ADD | 1B ADDC | 1B ORL dir, A 1C | 1B XRL dir, A 1C | 1B XRL dir, A 1C | 1B MOV dir, A 1C | 1B MOV dir, A 1C | 1B SUBB dir, A 1C | 1B CJNE A, R6, dir 2C | 1B CJNE A, R6, dir 2C | 1B XCHD A, R6, tc 1C | 1B XCHD A, R6, tc 1C | 1B MOV R6, A 1C | |
| F INC | 18 1C R7 | 1B DEC | 1B ADD | 1B ADDC | 1B ORL dir, A 1C | 1B XRL dir, A 1C | 1B XRL dir, A 1C | 1B MOV dir, A 1C | 1B MOV dir, A 1C | 1B SUBB dir, A 1C | 1B CJNE A, R7, dir 2C | 1B CJNE A, R7, dir 2C | 1B XCHD A, R7, tc 1C | 1B XCHD A, R7, tc 1C | 1B MOV R7, A 1C | |